Amendments to and Listing of the Claims:

This listing of claims replaces all prior versions and listings of claims in this application.

1. (Original) A method of replicating data associated with a plurality of transactions in a replication system including a plurality of nodes connected via communication media in a topology, each node including a database, at least some of the nodes being able to independently receive and post transactions, the method comprising:

(a) initiating and performing transactions to be executed in a database at an originating node; and

(b) replicating the transactions to at least one or more other nodes by:

(i) pausing each transaction being executed in the database at the originating node prior to a commit operation for the transaction,

(ii) assigning a ready to commit token to the transaction,

(iii) sending the ready to commit token to the one or more other nodes,

(iv) determining at the one or more other nodes whether the respective databases are prepared for a commit operation for the transaction corresponding to the ready to commit token, and, if so, sending back the ready to commit token to the originating node, and

(v) executing a commit operation at the database of the originating node only upon receipt from at least one of the other nodes of the ready to commit tokens originally sent from the originating node,

wherein step (b) is performed asynchronously with respect to other transactions that are initiated in step (a).

2. (Original) The method of claim 1 wherein the commit operation in step (b)(v) is performed only upon receipt from each of the one or more the other nodes of the ready to commit tokens originally sent from the originating node.

3. (Currently Amended) The method of claim 1 wherein the transactions ~~are initiated and performed in a~~ in step (a) are multi-threaded ~~manner~~.

4. (Original) A method of replicating data associated with a plurality of transactions in a data replication system including a plurality of nodes connected via communication media in a topology, each node including (i) a database, (ii) a replication engine which performs data replication functions, and (iii) an application which executes transactions and posts the transactions to the database, the application being independent of the replication engine, each transaction being one or more transaction steps or transaction operations, the method comprising:

(a) an application at a first node that initiating and performs transactions to be executed in a database at an originating node, the application pausing each transaction being executed in a source database at the first node prior to a commit operation for the transaction;

(b) a replication engine at the first node assigning a ready to commit token to the transaction in coordination with the application;

(c) the replication engine at the first node sending the ready to commit token to the second node;

(d) a replication engine at a second node determining whether a target database at the second node is prepared for a commit operation for the transaction corresponding to the ready to commit token, and, if so, sending back the ready to commit token to the first node; and

(e) the application at the first node executing a commit operation at the source database in coordination with the replication engine only upon receipt from the second node of the ready to commit token originally sent from the first node,

wherein the replication engine operates asynchronously with respect to the application until step (b) occurs.

5. (Currently Amended) The method of claim 4 wherein the transactions ~~are initiated and performed in a~~ <u>in step (a) are</u> multi-threaded ~~manner~~.

6. (Original) A method of performing dual writes for replicating transactions among plural databases located at different nodes, each transaction being one or more transaction steps or transaction operations, at least some of the transaction steps or transaction operations being update steps or operations, the method comprising:

(a) initiating a transaction at an originating node;

(b) inhibiting the dual write replication process from communicating transaction steps or operations of the transaction with one or more other nodes until an update step or operation occurs within the transaction at the originating node; and
(c) upon the occurrence of the update step or operation, performing the dual write replication process with respect to the one or more other nodes and sending with the update step or operation all transaction steps or operations for the transaction that have occurred prior to the update step or operation for the transaction, or prior to the previous update step or operation if a previous update step or operation existed for the transaction.

7. (Currently Amended) [[A]] The method ~~according to~~ of claim 6 further comprising:
(d) determining if the originating node needs to receive a data record from the one or more other nodes during the dual write replication process; and
(e) sending the data record to the originating node only if it is determined that the originating node needs the data record.

8. (Original) A method of performing dual writes for replicating transactions among plural databases located at different nodes, each transaction being one or more transaction steps or transaction operations, at least some of the transaction steps or transaction operations being update steps or operations which are performed only after database locking occurs, the method comprising:
(a) initiating a transaction at an originating node; and
(b) the dual write replication process causing database locking to occur at one or more other nodes only upon the occurrence of an update step or operation in the transaction at the originating node.

9. (Currently Amended) [[A]] The method ~~according to~~ of claim 8 further comprising:
(d) determining if the originating node needs to receive a data record from the one or more other nodes during the dual write replication process; and
(e) sending the data record to the originating node only if it is determined that the originating node needs the data record.